What does -1 mean in numpy reshape?

Asked 6 years, 8 months ago Active 27 days ago Viewed 267k times

407

203

A numpy matrix can be reshaped into a vector using reshape function with parameter -1. But I don't know what -1 means here.

For example:

```
a = numpy.matrix([[1, 2, 3, 4], [5, 6, 7, 8]])
b = numpy.reshape(a, -1)
```

The result of `b` is: `matrix([[1, 2, 3, 4, 5, 6, 7, 8]])`

Does anyone know what -1 means here? And it seems python assign -1 several meanings, such as: `array[-1]` means the last element. Can you give an explanation?

python numpy reshape numpy-ndarray

edited Dec 5 '19 at 7:19
kmario23
30.7k ●8 ●100 ●108

asked Sep 9 '13 at 3:25
user2262504
4,469 ●3 ●14 ●19

8 Answers

Active Oldest Votes

548

The criterion to satisfy for providing the new shape is that *'The new shape should be compatible with the original shape'*

numpy allow us to give one of new shape parameter as -1 (eg: (2,-1) or (-1,3) but not (-1, -1)). It simply means that it is an unknown dimension and we want numpy to figure it out. And numpy will figure this by looking at the *'length of the array and remaining dimensions'* and making sure it satisfies the above mentioned criteria

Now see the example.

```
z = np.array([[1, 2, 3, 4],
         [5, 6, 7, 8],
         [9, 10, 11, 12]])
z.shape
(3, 4)
```

Now trying to reshape with (-1) . Result new shape is (12,) and is compatible with original shape (3,4)

```
z.reshape(-1)
```

Now trying to reshape with (-1, 1) . We have provided column as 1 but rows as unknown . So we get result new shape as (12, 1).again compatible with original shape(3,4)

```
z.reshape(-1,1)
array([[ 1],
   [ 2],
   [ 3],
   [ 4],
   [ 5],
   [ 6],
   [ 7],
   [ 8],
   [ 9],
   [10],
   [11],
   [12]])
```

The above is consistent with `numpy` advice/error message, to use `reshape(-1,1)` for a single feature; i.e. single column

> Reshape your data using `array.reshape(-1, 1)` if your data has a **single feature**

New shape as (-1, 2). row unknown, column 2. we get result new shape as (6, 2)

```
z.reshape(-1, 2)
array([[ 1,  2],
   [ 3,  4],
   [ 5,  6],
   [ 7,  8],
   [ 9, 10],
   [11, 12]])
```

Now trying to keep column as unknown. New shape as (1,-1). i.e, row is 1, column unknown. we get result new shape as (1, 12)

```
z.reshape(1,-1)
array([[ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12]])
```

The above is consistent with `numpy` advice/error message, to use `reshape(1,-1)` for a single sample; i.e. single row

> Reshape your data using `array.reshape(1, -1)` if it contains a **single sample**

New shape (2, -1). Row 2, column unknown. we get result new shape as (2,6)

```
z.reshape(2, -1)
array([[ 1,  2,  3,  4,  5,  6],
   [ 7,  8,  9, 10, 11, 12]])
```

```
z.reshape(3, -1)
array([[ 1,  2,  3,  4],
   [ 5,  6,  7,  8],
   [ 9, 10, 11, 12]])
```

And finally, if we try to provide both dimension as unknown i.e new shape as (-1,-1). It will throw an error

```
z.reshape(-1, -1)
ValueError: can only specify one unknown dimension
```

edited Jul 15 '19 at 5:46
The Red Pea
9,127 ●7 ●60 ●99

answered Feb 28 '17 at 13:48
Julu Ahamed
5,631 ●1 ●10 ●10

7 This answer contains a lot of examples but doesn't lay out what -1 does in plain English. When reshaping an array, the new shape must contain the same number of elements as the old shape, meaning the products of the two shapes' dimensions must be equal. When using a -1, the dimension corresponding to the -1 will be the product of the dimensions of the original array divided by the product of the dimensions given to `reshape` so as to maintain the same number of elements. – BallpointBen Apr 30 '18 at 21:50

1 In my opinion the accepted answer and this answer are both helpful, whereas the accepted answer is more simple, I prefer the simpler answer – cloudscomputes Aug 23 '18 at 2:34

1 How is the shape (12, 1) "compatible" with shape (3,4)? – Vijender Feb 10 '19 at 23:12

Perfect answer! +1 – Kaushal28 Apr 6 '19 at 17:51

1 @Vijender I guess it means the same number of elements but different axis - i.e. 12x1 == 3x4? – David Waterworth Apr 21 at 0:19

79

Used to reshape an array.

Say we have a 3 dimensional array of dimensions 2 x 10 x 10:

```
r = numpy.random.rand(2, 10, 10)
```

Now we want to reshape to 5 X 5 x 8:

```
numpy.reshape(r, shape=(5, 5, 8))
```

will do the job.

Note that, once you fix first dim = 5 and second dim = 5, you don't need to determine third dimension. To assist your laziness, python gives the option of -1:

```
numpy.reshape(r, shape=(5, 5, -1))
numpy.reshape(r, shape=(50, -1))
```

will give you an array of shape = (50, 4)

You can read more at http://anie.me/numpy-reshape-transpose-theano-dimshuffle/

edited Apr 23 '18 at 12:37
Xan
61.1k ● 13 ● 132 ● 158

answered Mar 22 '17 at 11:35
Anuj Gupta
4,173 ● 5 ● 27 ● 43

59

According to `the documentation` :

> newshape : int or tuple of ints
>
> The new shape should be compatible with the original shape. If an integer, then the result will be a 1-D array of that length. One shape dimension can be **-1. In this case, the value is inferred from the length of the array and remaining dimensions.**

answered Sep 9 '13 at 3:27
falsetru
283k ● 39 ● 529 ● 502

In this case, the value is inferred to be [1, 8]. And 8 is the total number of matrix a. right? – user2262504 Sep 9 '13 at 3:35

@user2262504, I'm not sure. I think the value inferred is `[8]` because the documentation say so (`1-D array`). Try `numpy.reshape(a, [8])` . It yields same result with `numpy.reshape(a, [1,8])` for the matrix. – falsetru Sep 9 '13 at 3:55

3 -1 lets numpy determine for you the unknown number of columns or rows in the resulting matrix. Note: the unknown should be either columns or rows, not both. – Gathide May 10 '17 at 10:07

15

numpy.reshape(a,newshape,order{}) check the below link for more info. https://docs.scipy.org/doc/numpy/reference/generated/numpy.reshape.html

for the below example you mentioned the output explains the resultant vector to be a single row. (-1) indicates the number of rows to be 1. if the

```
a = numpy.matrix([[1, 2, 3, 4], [5, 6, 7, 8]])
b = numpy.reshape(a, -1)
```

output:

matrix([[1, 2, 3, 4, 5, 6, 7, 8]])

this can be explained more precisely with another example:

output:(is a 1 dimensional columnar array)

array([[0],

```
 [1],
 [2],
 [3],
 [4],
 [5],
 [6],
 [7],
 [8],
 [9]])
```

b = np.arange(10).reshape((1,-1))

output:(is a 1 dimensional row array)

array([[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]])

edited Jan 2 '17 at 17:19
Panther
3,008 ●9 ●22 ●46

answered Jan 2 '17 at 16:41
Dinesh Kumar
196 ●1 ●5

12

It is fairly easy to understand. The "-1" stands for "unknown dimension" which can should be infered from another dimension. In this case, if you set your matrix like this:

```
a = numpy.matrix([[1, 2, 3, 4], [5, 6, 7, 8]])
```

Modify your matrix like this:

```
b = numpy.reshape(a, -1)
```

It will call some deafult operations to the matrix a, which will return a 1-d numpy array/martrix.

However, I don't think it is a good idea to use code like this. Why not try:

```
b = a.reshape(1,-1)
```

It will give you the same result and it's more clear for readers to understand: Set b as another shape of a. For a, we don't how much columns it should have(set it to -1!), but we want a 1-dimension array(set the first parameter to 1!).

answered Feb 27 '17 at 1:56
F0rge1cE
121 ●1 ●2

9

```
(userDim1, userDim2, ..., -1) -->>

(userDim1, userDim1, ..., TOTAL_DIMENSION - (userDim1 + userDim2 + ...))
```

answered Dec 8 '18 at 15:26
Shayan Amani
3,238 ● 26 ● 31

This is the answer in English I was looking for, plain and simple. i.e you give the your design preference, let numpy work out the remaining math :) – Sumanth Lazarus Aug 5 '19 at 12:11

6

It simply means that you are not sure about what number of rows or columns you can give and you are asking numpy to suggest number of column or rows to get reshaped in.

numpy provides last example for -1
https://docs.scipy.org/doc/numpy/reference/generated/numpy.reshape.html

check below code and its output to better understand about (-1):

CODE:-

```
import numpy
a = numpy.matrix([[1, 2, 3, 4], [5, 6, 7, 8]])
print("Without reshaping  -> ")
print(a)
b = numpy.reshape(a, -1)
print("HERE We don't know about what number we should give to row/col")
print("Reshaping as (a,-1)")
print(b)
c = numpy.reshape(a, (-1,2))
print("HERE We just know about number of columns")
print("Reshaping as (a,(-1,2))")
print(c)
d = numpy.reshape(a, (2,-1))
print("HERE We just know about number of rows")
print("Reshaping as (a,(2,-1))")
print(d)
```

OUTPUT :-

```
Without reshaping  ->
[[1 2 3 4]
 [5 6 7 8]]
HERE We don't know about what number we should give to row/col
Reshaping as (a,-1)
[[1 2 3 4 5 6 7 8]]
HERE We just know about number of columns
Reshaping as (a,(-1,2))
[[1 2]
 [3 4]
 [5 6]
 [7 8]]
```

```
[[1 2 3 4]
 [5 6 7 8]]
```

answered Nov 27 '19 at 13:12
lonewolf
151 ● 2 ● 2

2

```
import numpy as np
x = np.array([[2,3,4], [5,6,7]])

# Convert any shape to 1D shape
x = np.reshape(x, (-1)) # Making it 1 row -> (6,)

# When you don't care about rows and just want to fix number of columns
x = np.reshape(x, (-1, 1)) # Making it 1 column -> (6, 1)
x = np.reshape(x, (-1, 2)) # Making it 2 column -> (3, 2)
x = np.reshape(x, (-1, 3)) # Making it 3 column -> (2, 3)

# When you don't care about columns and just want to fix number of rows
x = np.reshape(x, (1, -1)) # Making it 1 row -> (1, 6)
x = np.reshape(x, (2, -1)) # Making it 2 row -> (2, 3)
x = np.reshape(x, (3, -1)) # Making it 3 row -> (3, 2)
```

answered Apr 10 at 5:40
Sherzod
1,357 ● 2 ● 11 ● 27

Highly active question. Earn 10 reputation in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.